ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ

«Университет «Дубна»

ИНСТИТУТ СИСТЕМНОГО АНАЛИЗА И УПРАВЛЕНИЯ

Кафедра системного анализа и управления

КУРСОВАЯ РАБОТА

по дисциплине

«Программированиена языке высокого уровня»

Реализация игры «Тетрис»

ТЕМА: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1013

Выполнил: студент группы \_\_\_\_\_\_\_\_\_\_\_\_

Ильин Михаил Алексеевич

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

(Ф.И.О.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента)

**Руководители**:

по дисциплине ОИТ

доцент Мурадян Аруцюн Ваникович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

по дисциплине ПЯВУ

Ст. преп. Самойлов Юрий Евгеньевич

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись руководителя)

Дубна, 2021

Оглавление

[Введение 3](#_Toc73078333)

[Задачи работы 4](#_Toc73078334)

[Теоретическая часть игры 5](#_Toc73078335)

[Практическая часть игры 6](#_Toc73078336)

[Выводы 11](#_Toc73078337)

[Список литературы 13](#_Toc73078338)

[Приложение 14](#_Toc73078339)

# Введение

Идея «Тетриса» родилась у Алексея Пажитнова в 1984 году после знакомства с головоломкой американского математика Соломона Голомба *Pentomino Puzzle*. Суть этой головоломки была довольно проста и до боли знакома любому современнику: из нескольких фигур нужно было собрать одну большую. Алексей решил сделать компьютерный вариант пентамино [1]. В таком исполнении «Тетрис» знают большинство из нас. На поле 10×20 нужно выставлять падающие сверху фигуры внизу как можно компактнее. Если ряд заполнен – вы получаете очки. Однако победить в этой игре нельзя, можно лишь накапливать их на свой счет. Чем больше рядов одновременно заполнилось, тем больше очков вы получаете. Однако и темп игры ускоряется – фигуры падают быстрее. Игрок, получивший больше всего очков, является лидером. То есть, цель игры – получить больше очков, чем у лидера.

После переноса игры на *PC*, распространялся «Тетрис» на набиравших тогда популярность дискетах путем банального копирования у друзей и за две недели игра расползлась по всей Москве, а потом и по всему СССР. В данной курсовой работе представлена реализация этой популярной игры, развивающей скорость реакции и логическое мышление.

# Задачи работы

Цель: разработать и реализовать игру «Тетрис» на *Microsoft.NET* на языке *C*#.

Исходные данные: правила игры «Тетрис» и описание ее работы, официальная документация по языку *C*#.

Ожидаемый результат: реализованная на языке *C*# программа, которая дает возможность пользователю играть в «Тетрис».

# Теоретическая часть игры

Случайные фигурки тетрамино (состоящие из 4 закрашенных клеток) падают сверху в прямоугольной карте шириной 10 и высотой 20 клеток. В полёте игрок может поворачивать фигурку на 90° и двигать её по горизонтали. Также можно «сбрасывать» фигурку, то есть ускорять её. Фигурка летит до тех пор, пока не наткнётся на другую фигурку либо на дно карты. Если при этом заполнился горизонтальный ряд из 10 клеток, он пропадает и всё, что выше него, опускается на одну клетку вниз. Темп игры постепенно ускоряется. Игра заканчивается, когда новая фигурка не может поместиться в стакан. Игрок получает очки за каждый заполненный ряд, поэтому его задача — заполнять ряды, не заполняя саму карту (по вертикали) как можно дольше, чтобы таким образом получить как можно больше очков.

Начисление очков в разных версиях «Тетриса» довольно разнообразное: в этой версии за 1 ряд дается 4 очка, 2 ряда – 20 очков, 3 ряда – 84 очка, 4 ряда (то есть сделать Тетрис) – 364 очка.

# Практическая часть игры

Реализация всей курсовой работы делится на несколько этапов:

1. Создание титульного листа. (Рис. 1).

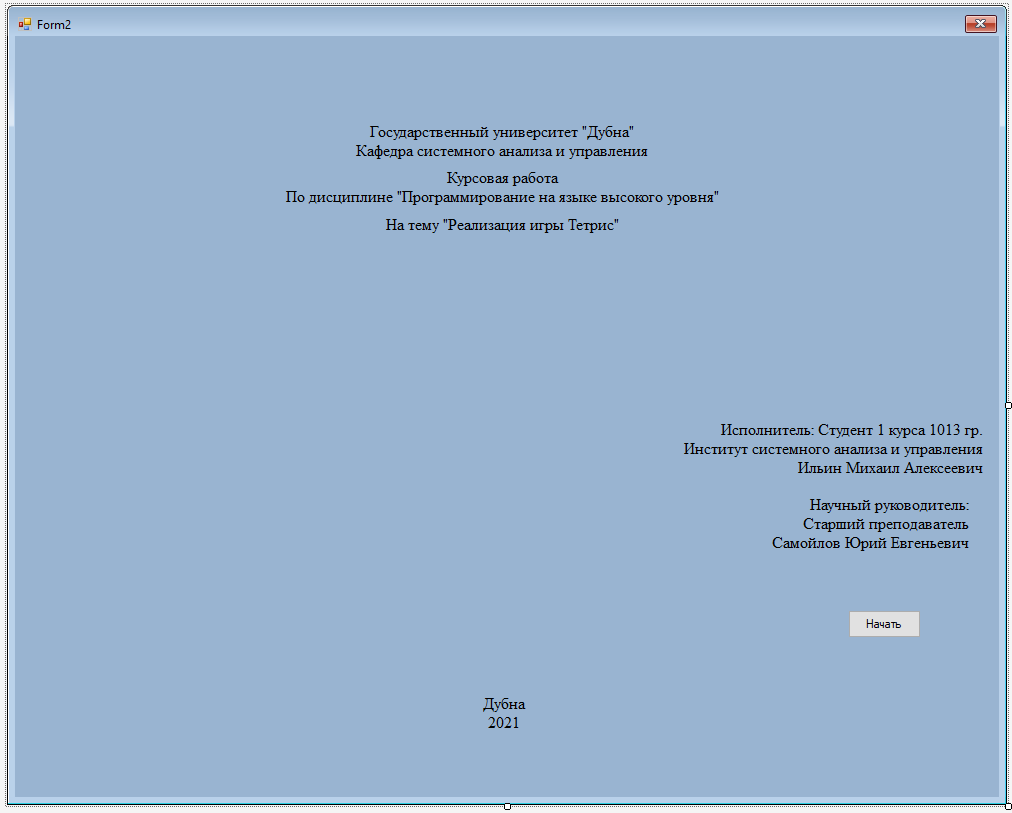


Рис. 1. Титульный лист

1. Создание дизайна приложения с игрой с минималистичным и простым интерфейса (см. Приложение, Рис. 1).
2. Создание игрового поля, а также управляемого объекта (см. Приложение, Рис. 2).
3. Написание кода программы на языке *C#*.

Для реализации титульного листа в большей степени использовались элементы «*label*» – Элементы управления *Windows Forms* используются для вывода текста или изображений, которые не могут быть изменены пользователем [2].

При запуске программы пользователь видит (Рис. 1) титульный лист с кнопкой, которая запускает форму регистрации нового пользователя.

При нажатии на кнопку «начать» открывается основная форма, на которой и будет реализовываться игра. На основной форме представлены:

* *menuStrip*, в котором находятся кнопки «Новая игра», «Результаты», «Пауза», «Справка»;
* панель с *label*, на которых представлены счет и убранные линии.

Рассмотрим основные части кода (Рис. 2); (Рис. 3); (Рис. 4):

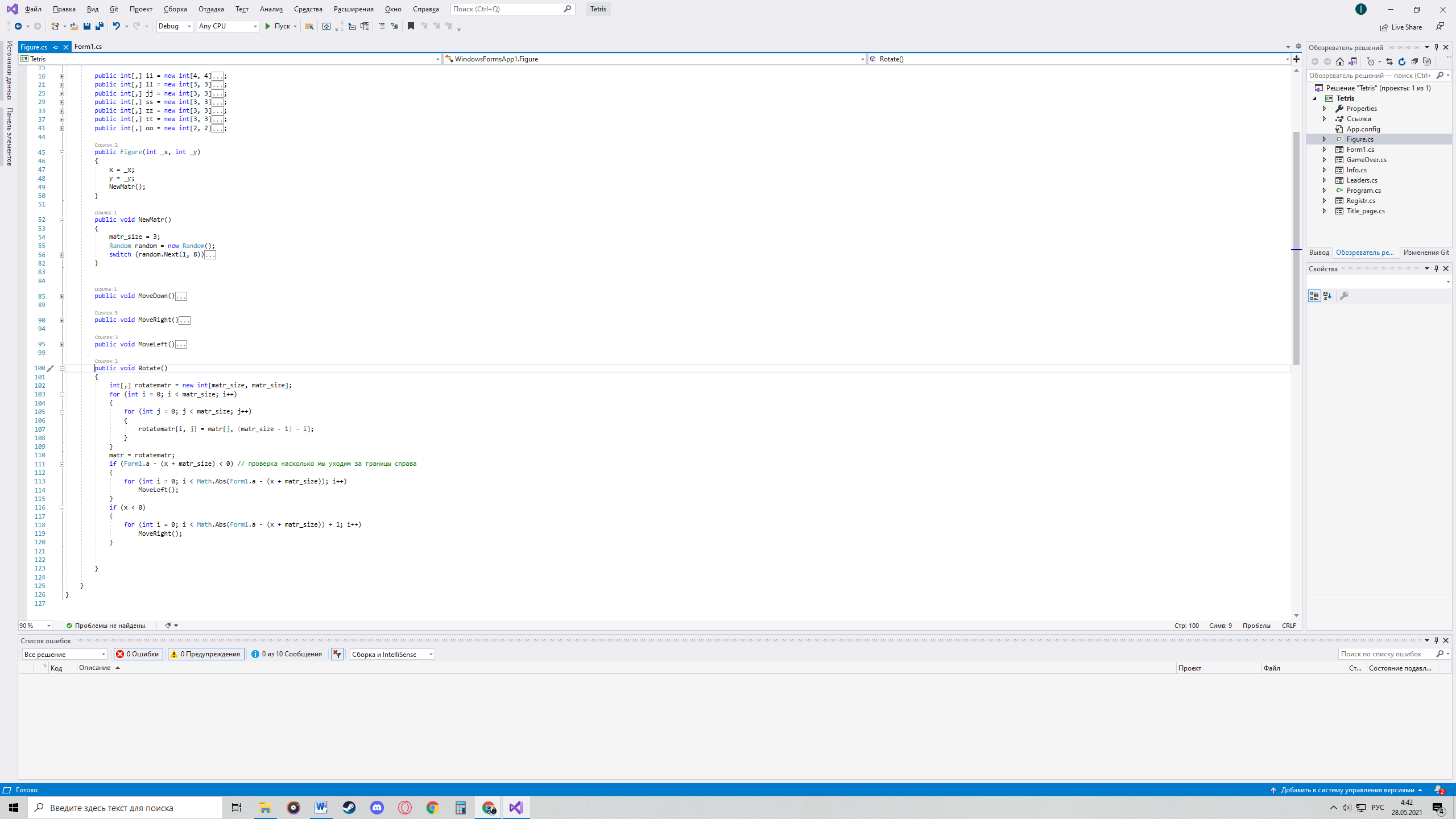


Рис 2. Часть кода, отвечающая за поворот фигуры

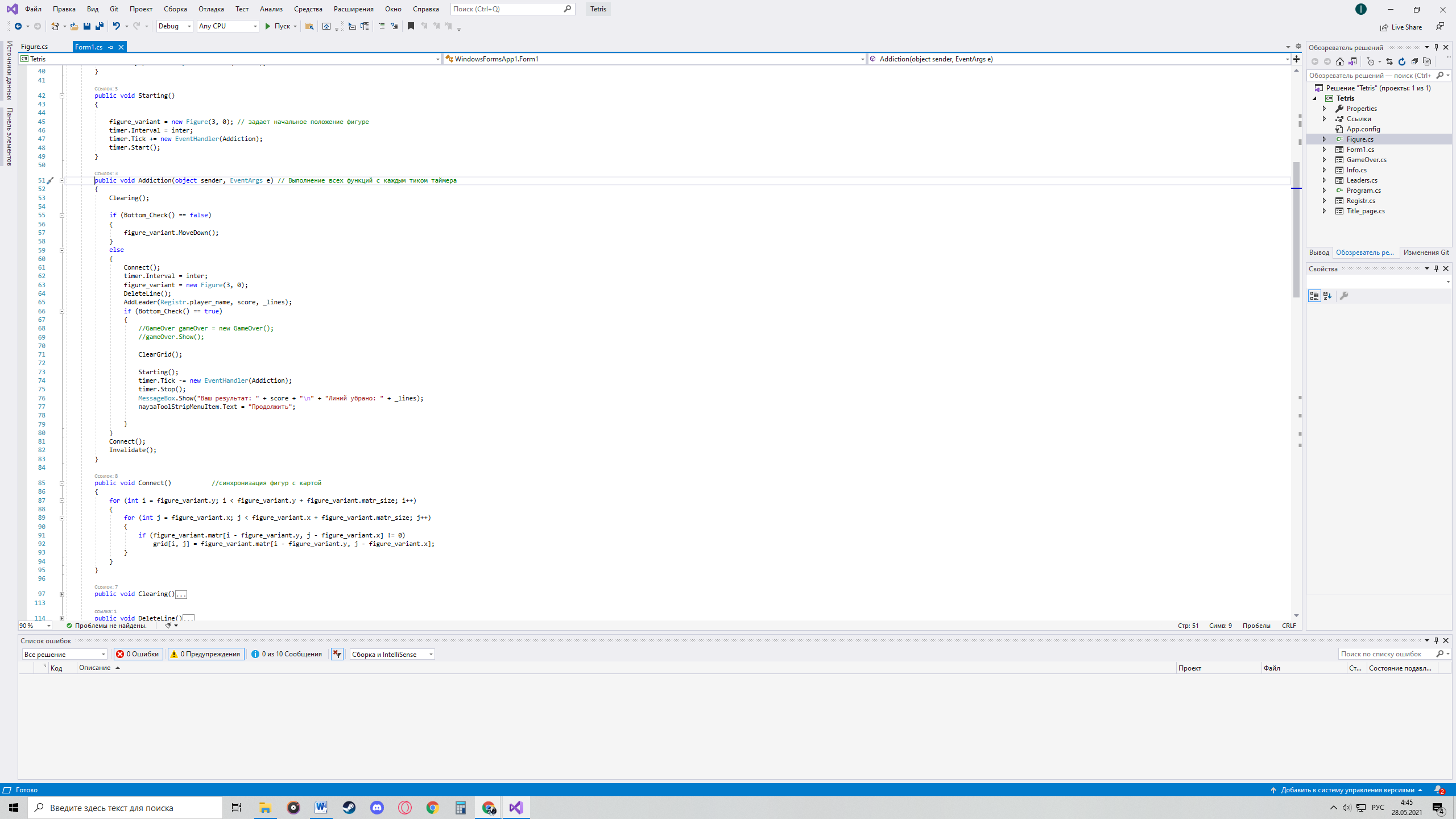


Рис. 3 Часть кода, на которой прописано выполнение всех функций с каждым тиком таймера

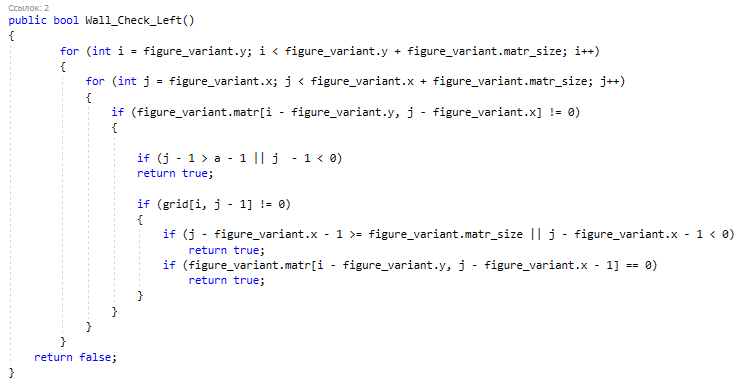


Рис. 4. Часть кода, на которой прописано одно из нескольких условий проверки движения фигуры.

Первее других двух, вызывается метод *Addiction*(), он же и является самым главным во всей программе. Этот метод привязан к тику таймера и срабатывает каждые *inter* секунд. Первым делом очищается значение фигуры предыдущего тика и записывается в новое место (*y* + 1), таким образом она падает вниз. Далее мы должны проверить, а можем ли мы вообще двигаться. Если да, то двигаемся до ближайшего блока под фигурой, и, как только доходим, метод *Bottom\_Check* возвращает значение *true*. Это означает, что нынешнюю фигуру мы двигать не можем, и поэтому фиксируем ее и создаем новую.

Это продолжается до тех пор, пока количество фигур под нами не станет настолько велико, что они будут создавать «дно» карты уже при ее появлении, а это означает конец игры. С помощью функций *Connect* и *Invalidate* мы каждые *inter* секунд проверяем привязку фигуры к карте и отрисовываем ее, *Invalidate* помогает сделать это плавно.

Одна из основных функций «Тетриса» – это функция поворота. Она представлена в методе *Rotate*. Само вращение сделать просто: мы берем нашу исходную фигуру (для кода программы она является матрицей), и вращаем ее вокруг центра сетки матрицы.

Но вращать фигуру не всегда можно, поэтому перед вращением мы исключим самую простую ошибку – выход при повороте за границы карты. Если же все таки она выходит, мы считаем, на сколько она за картой и двигаем ее соответственно в противоположную сторону на это количество.

Ну и метод проверки *Wall\_Check\_Left/Right* работает в постоянном режиме. Он все время проверяет, чтобы мы не двигали фигуру за карту и мы не могли сдвигать фигуру в другую фигуру. Фигура, как я уже описывал выше, для программы является матрицей. В зависимости от варианта фигуры в матрицу записывается число от 1 до 7, означающие их цвета.

Соответственно во все нулевые участки мы можем двигать фигурку.

# Выводы

В ходе выполнения данной курсовой работы была создана программа, повторяющая оригинальный «Tetris» Программу можно улучшить следующими способами: добавить новые уровни игры, отличающиеся сложностью, реализовать возможность играть вдвоем состязательном в режиме.

На рисунке ниже показана блок-схема реализованной программы.

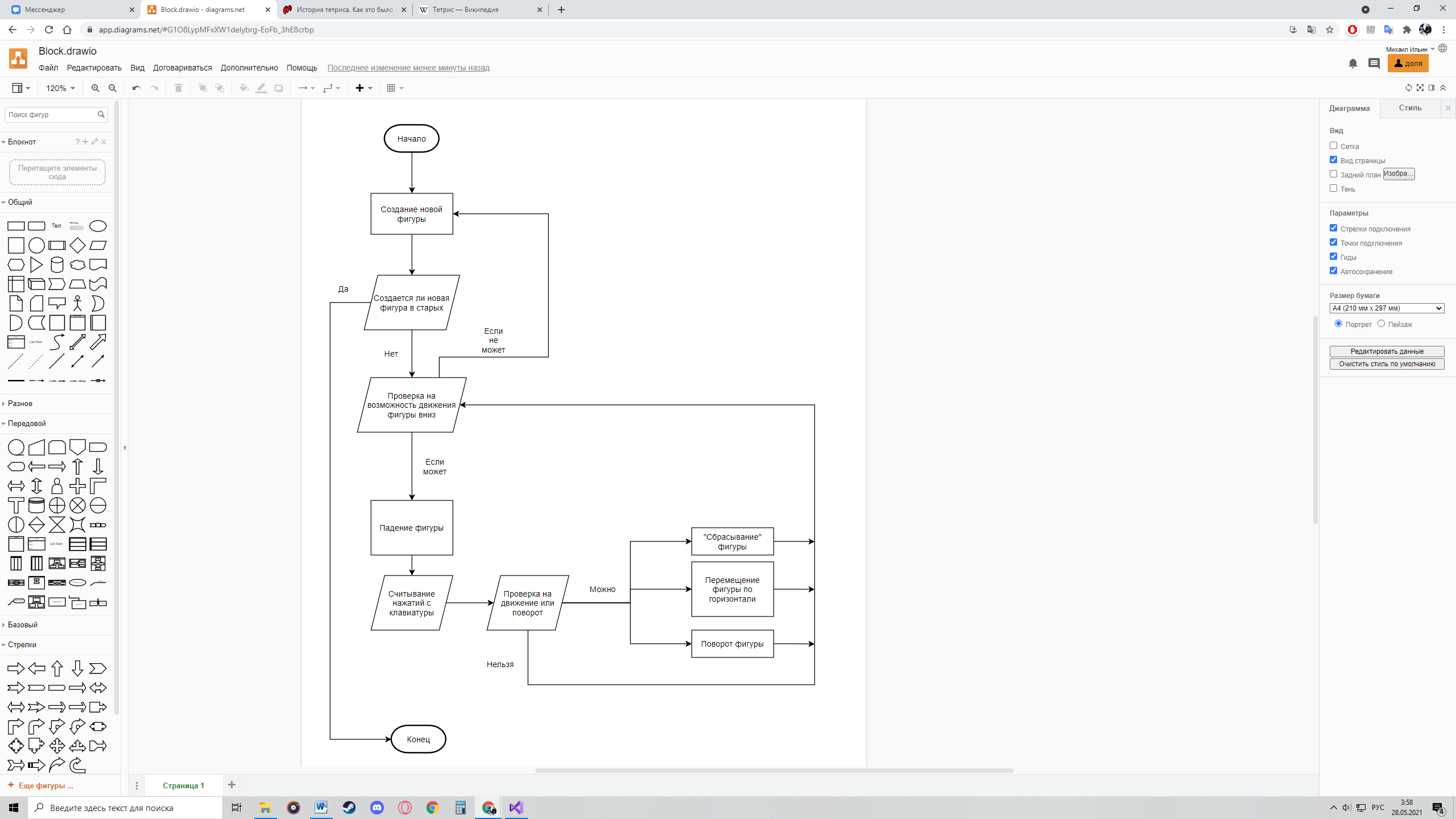
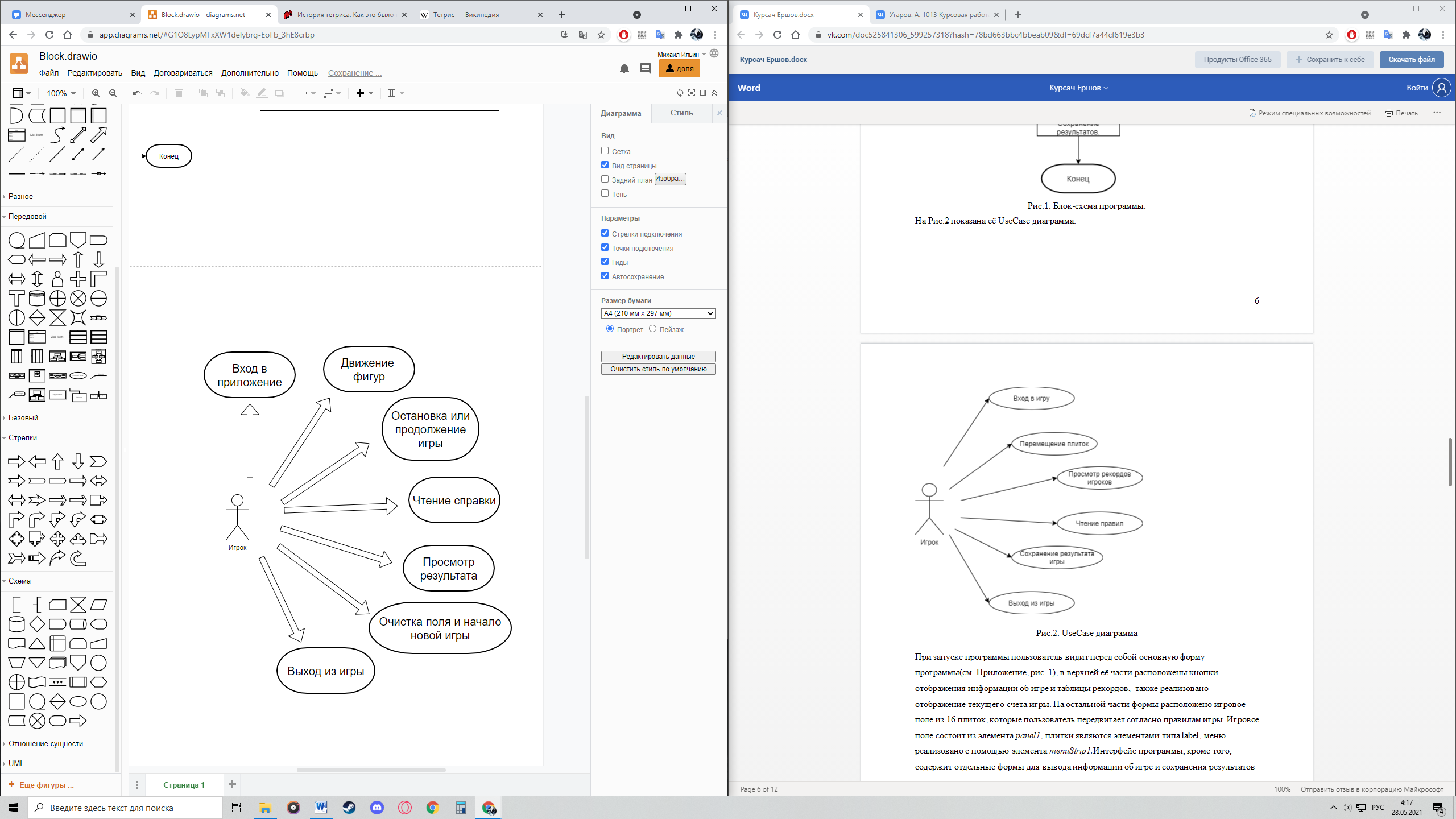


Рис. 1. Блок-схема программы.

На Рис.2 показана её UseCase диаграмма.



На Рис.2. UseCase диаграмма.

# Список литературы

1. История тетриса. Как это было на самом деле [Электронный ресурс] – Режим доступа <https://www.igromania.ru/article/11612/Istoriya_tetrisa_Kak_yeto_bylo_na_samom_dele.html> – Яз. Рус. (дата обращения 15.05.2021)
2. Элемент управления *Label* (*Windows* *Form*) [Электронный ресурс] — Режим доступа <https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.label?view=net-5.0> свободный. — Загл. с экрана. — Яз. Рус. (дата обращения 15.05.2021)

# Приложение

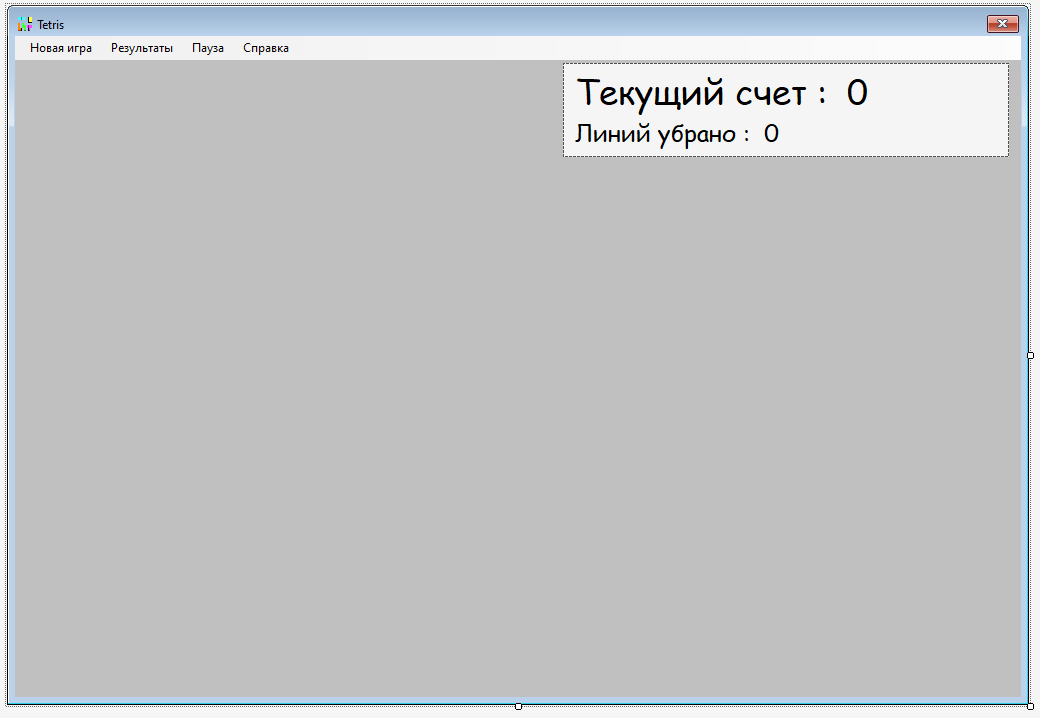


Рис. 1. Дизайн приложения



Рис. 3. Игровое поле с объектом